IRM PRESS
701 E. Chocolate Avenue, Hershey PA 17033-1117, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.irm-press.com
ITB9094

Chapter V

Analysis of Learner Performance on a Tutoring System for Java

Henry H. Emurian
University of Maryland–Baltimore County, USA

Jingli Wang
University of Maryland–Baltimore County, USA

Ashley G. Durham[1]
Centers for Medicare & Medicaid Services (CMS), USA

ABSTRACT

This chapter presents a teaching methodology, programmed instruction, that provides a series of interactive and cumulative learning experiences that teach a student how to understand and write a simple Java Applet. Fine-grain performance records of three students' interactions with the tutoring system show the individual patterns of skill acquisition and retention over five successive observational occasions. The tutoring system is also used as the first technical exercise in a course entitled "Graphical User Interface Systems Using Java." Performance and self-reported ratings of programming confidence by 17 graduate students show the benefit of programmed instruction to generate a history of competency and confidence in all students. The positive initial experience prepares and motivates information systems students for the presentation and mastery of advanced programming techniques.

INTRODUCTION

The purpose of this chapter is to demonstrate student learning and retention on components of a Web-based tutoring system that teaches a Java Applet within the framework of a competency model of instructional design: programmed instruction. The objective of the tutoring system is to provide an initial and positive learning experience for information systems students who may lack a background in computer programming. The learning experience is intended to foster a student's tested competency in writing and understanding a simple Java Applet. The teaching objective is accomplished within a learning framework that is intended to generate a history of symbol manipulation and understanding within the context of a tutoring system that offers structured rehearsal to a criterion of mastery. The use of the system is also intended to produce a positive affective experience for the learner. Finally, the tutoring system is designed to be used only as the introductory laboratory in a technically oriented course in which subsequent classes are taught by lecture, supervised laboratory, and collaboration formats.

The instructional model is based upon programmed instruction, which combines teaching, practice, and competency testing within a single conceptual framework, and it assures the achievement of a criterion of mastery at the level of the individual student (e.g., Anger et al., 2001; Holland, 1960; Scriven, 1969; Skinner, 1958). The examples to be presented in this chapter build upon our previous work (Emurian et al., 2000; Emurian & Durham, 2001; Emurian & Durham, in press) that provides a pedagogical context and rationale for the adoption of programmed instruction approaches for technical training in information technology.

BACKGROUND

Computer programming appears in the recommendations for core courses in several curriculum guidelines for the academic discipline of information systems.[2] The activity of constructing and understanding a computer program has been extensively investigated in the literature, and the complexity of current end-user applications suggests the relevance of that research to general issues of approaches to information technology education and software training (Bannert & Reimann, 2000). Early work in this area of research emphasized the conditions that promote a learner's exploratory mastery of logical constructions and flow of control (Papert, 1980). Research perspectives related to classroom teaching and student learning of computer programming were later addressed by Mayer (1988), and the activity of constructing and comprehending a computer program and command sequences continues to be investigated (e.g., Altmann, 2001; Campbell, Brown, & DiBello, 1992; Sohn & Doane, 1997; Soloway, 1985; Van Merrienboar & Paas, 1990). It is the case, perhaps, that the impact of this stream of research has yet to be realized in the classroom, as evidenced by the adoption of new teaching methods by computer programming instructors.

A similar situation may exist with respect to investigations of training for end-user computing applications. For example, some approaches to the development of interface software tools are relevant to suggesting prerequisite competencies for learning an advanced tool (Myers, Hudson, & Pausch, 2000), and learning effective strategies may help users to optimize performance on software tools (Bhavnani & John, 2000). Moreover, the training literature has begun to show appreciation of the potential impact on learning of the trainer's personal style (Compeau, 2002) and the motivation of learners to acquire new skills (Ryan,